CLAIMS

1. A method for protecting the secrecy and integrity of data stored on a non-volatile storage medium, the method comprising:

receiving a block of data for storage on the non-volatile storage medium;

generating at least one piece of meta-data relating to the block of data;

calculating a first cryptographic hash of at least a portion of the block of data;

calculating a second cryptographic hash of the meta-data;

encrypting the block of data and encrypting the meta-data to form one or more uniform blocks of encrypted data;

storing a cryptographic key in a substantially secret storage medium, the key being operable to decrypt the one or more uniform blocks of encrypted data;

storing the one or more uniform blocks of encrypted data on the non-volatile storage medium.

2. A method as in claim 1, further comprising:

receiving a request for the block of data;

retrieving the cryptographic key from the secret storage medium;

retrieving the one or more uniform blocks of encrypted data from the non-volatile storage medium;

decrypting the one or more uniform blocks of encrypted data to yield a decrypted version of the block of data and a decrypted version of the meta-data;

calculating a third cryptographic hash by hashing the decrypted version of the block of data;

calculating a fourth cryptographic hash by hashing the decrypted version of the meta-data;

comparing the third cryptographic hash with the first cryptographic hash; and

granting the request for the block of data if the third cryptographic hash is equal to the first cryptographic hash.

3. A method as in claim 1, wherein the one or more uniform blocks of data are stored on the non-volatile storage medium in a log-structured file.

4. A method as in claim 1, further comprising:

generating a hierarchical location map for use in locating the one or more uniform blocks of encrypted data on the non-volatile storage medium, the location map comprising one or more nodes, a first node of which contains the first cryptographic hash and an indicator specifying the location on the non-volatile storage medium of the portion of the block of data to which the first cryptographic hash corresponds.

5. A method as in claim 4, further comprising:

computing a third cryptographic hash by hashing said first node;

encrypting said first node;

storing said first node on the non-volatile storage medium.

6. A method as in claim 5, further comprising:

storing the third cryptographic hash in a second node of said hierarchical location map;

storing in said second node of said hierarchical location map an indicator specifying the location on the non-volatile storage medium of the first node.

7. A method of managing the storage of a plurality of data blocks on a storage medium, the method comprising:

storing the plurality of data blocks on the storage medium;

generating a hierarchical location map for locating individual ones of said plurality of blocks, the hierarchical location map including a plurality of nodes, wherein a first node type includes:

one or more hash values of subordinate nodes or data blocks; and

one or more location indicators specifying the location at which subordinate nodes or data blocks are stored on said storage medium; and

wherein a second node type includes:

a hash value of a subordinate node;

a location indicator specifying the location at which the subordinate node is stored on said storage medium;

a cryptographic key for decrypting one or more subordinate nodes.

8. A method as in claim 7, in which the plurality of data blocks are stored on the storage medium in a log-structured file.

9. A method as in claim 7, in which the second node type further includes:

an indicator of the type of cryptographic algorithm used to encrypt the one or more subordinate nodes.

10. A method as in claim 9, in which the location map contains at least a first and second node of the second node type, and in which the first and second nodes of the second node type contain different cryptographic keys, and indicators specifying different cryptographic algorithms.

11. A secure database system, the system comprising:

an interface module for receiving data to be stored in the secure database;

a data management module for generating indexing information relating to the data to be store in the secure database;

a validation module operable to compute a hash of at least a portion of the data to be stored in the secure database and to compute a hash of at least a portion of the indexing information;

a cryptographic module operable to encrypt at least a portion of the data to be stored in the secure database and to encrypt at least a portion of the indexing information;

a storage medium operable to receive chunks of encrypted data and encrypted indexing information, and to store the chunks.

12. A data storage system, comprising:

a bulk storage device;

a trusted processing environment;

a computer-implemented database management system, comprising:

computer code for authenticating an application program that attempts to interface with the database management system;

computer code for receiving requests to store or retrieve data from an authenticated application program;

computer code for generating indexing information pertaining to data received from the authenticated application program;

computer code for generating hash values by hashing the data received from the authenticated application program, and for hashing the indexing information pertaining to the data received from the authenticated application program;

computer code for encrypting the data received from the authenticated application program, and for encrypting the indexing information pertaining to the data received from the authenticated application program;

computer code for storing the encrypted data and the encrypted indexing information on the bulk storage medium;

computer code for retrieving the encrypted data and the encrypted indexing information from the bulk storage medium;

computer code for decrypting the encrypted data and the encrypted indexing information;

computer code for authenticating the decrypted data and the decrypted indexing information using said hash values;

wherein the computer codes for said database management system are loaded into the trusted processing environment, and are used to manage

the storage and retrieval of data received from the authenticated application program.

13. A system as in claim 12, in which the trusted processing environment comprises an integrated circuit contained in a tamper-resistant case, and in which the integrated circuit includes:

a volatile memory unit for storing at least a portion of the computer codes of said computer-implemented database management system.

14. A computer program product for managing data received from an application program, the computer program product including:

computer code for receiving requests to store or retrieve data from the application program;

computer code for generating indexing information pertaining to data received from the application program;

computer code for generating hash values by hashing the data received from the application program, and for hashing the indexing information pertaining to the data received from the application program;

computer code for encrypting the data received from the application program, and for encrypting the indexing information pertaining to the data received from the application program;

computer code for storing the encrypted data and the encrypted indexing information on a storage medium;

computer code for retrieving the encrypted data and the encrypted indexing information from the storage medium;

computer code for decrypting the encrypted data and the encrypted indexing information;

computer code for authenticating the decrypted data and the decrypted indexing information using said hash values; and

a computer readable storage medium for containing said computer codes.

15. A computer program product as in claim 14, in which the computer readable medium is one of: CD-ROM, DVD, MINIDISC, floppy disk, magnetic tape, flash memory, ROM, RAM, system memory, hard drive, optical storage, and a data signal embodied in a carrier wave.

16. A method for protecting a database system from external analysis and attack, the method comprising:

disguising the size of data blocks targeted for storage on a storage medium;

disguising the frequency with which data blocks are written to the storage medium;

and disguising the location at which data blocks are stored on the storage medium.

17. A method as in claim 16, in which disguising the size of data blocks includes padding the data blocks with additional data.

18. A method as in claim 17, in which the data blocks are padded such that each block falls with predefined size limitations.

19. A method as in claim 17, in which the data blocks are padded such that each block is of substantially the same size as the other data blocks.

20. A method as in claim 16, in which disguising the frequency with which data blocks are written to the storage medium includes:

writing data blocks to the storage medium at a substantially constant frequency.

21. A method as in claim 16, in which disguising the frequency with which data blocks are written to the storage medium includes:

writing data blocks to the storage medium at a frequency that varies according to a predefined pattern.

22. A method as in claim 16, in which disguising the location at which data blocks are stored on the storage medium includes:

relocating data blocks that were previously stored on the storage medium to new locations on the storage medium when said data blocks are rewritten or updated.

23. A method as in claim 22, in which the new locations are selected quasi-randomly.